Isotropic Remeshing of Surfaces: a Local Parameterization Approach

Vitaly Surazhsky, Pierre Alliez, Craig Gotsman

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Isotropic Remeshing of Surfaces: a Local Parameterization Approach

Vitaly Surazhsky — Pierre Alliez — Craig Gotsman

N° 4967

Octobre 2003

THÈME 2

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--4967--FR+ENG

Isotropic Remeshing of Surfaces: a Local Parameterization Approach

Vitaly Surazhsky* , Pierre Alliez† , Craig Gotsman‡

Thème 2 —Génie logiciel
et calcul symbolique
Projet Geometrica

Rapport de recherche n° 4967 —Octobre 2003 — pages

Abstract: We present a method for isotropic remeshing of arbitrary genus surfaces. The method is based on a mesh adaptation process, namely, a sequence of local modifications performed on a copy of the original mesh, while referring to the original mesh geometry. The algorithm has three stages. In the first stage the required number or vertices are generated by iterative simplification or refinement. The second stage performs an initial vertex partition using an area-based relaxation method. The third stage achieves precise isotropic vertex sampling prescribed by a given density function on the mesh. We use a modification of Lloyd's relaxation method to construct a weighted centroidal Voronoi tessellation of the mesh. We apply these iterations locally on small patches of the mesh that are parameterized into the 2D plane. This allows us to handle arbitrary complex meshes with any genus and any number of boundaries. The efficiency and the accuracy of the remeshing process is achieved using a patch-wise parameterization technique.

Key-words: Surface mesh generation, isotropic triangle meshing, centroidal Voronoi tessellation, local parameterization.

* Technion, Haifa, Israel
† INRIA Sophia-Antipolis
‡ Technion, Haifa, Israel

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 —Télécopie : +33 4 92 38 77 65

Remaillage isotrope de surfaces utilisant une paramétrisation locale

Résumé : Cet article décrit une méthode de remaillage isotrope de surfaces triangulées. L'approche repose sur une technique d'adaptation locale du maillage. L'idée consiste à opérer une séquence d'opérations élémentaires sur une copie du maillage original, tout en faisant référence au maillage original pour la géométrie. L'algorithme comporte trois étapes. La première étape ramène la complexité du maillage au nombre de sommets désiré par raffinement ou décimation itérative. La seconde étape opère une première répartition des sommets via une technique de relaxation optimisant un équilibrage local des aires sur les triangles. La troisième étape opère un placement isotrope des sommets via une relaxation de Lloyd pour construire une tessellation de Voronoi centrée. Les itérations de relaxation de Lloyd sont appliquées localement dans un espace paramétrique 2D calculé à la volée sur un sous-ensemble de la triangulation originale de telle que sorte que les triangulations de complexité et de genre arbitraire puissent être efficacement remaillées.

Mots-clés : Maillage de surfaces, maillage triangulaire isotrope, diagrammes de Voronoi centrés, paramétrisation locale.

1 Introduction

Mesh generation has received a great deal of attention by researchers in various areas ranging from computer graphics through numerical analysis to computational geometry. Quality mesh generation amounts to finding a partition of a domain by elements—typically, triangles or quads. The shape, angles or size of these elements must match certain criteria (see [1, 2]). In most cases the boundary of the domain is given, as well as an importance map that must be discretized. The problem of surface *remeshing*, being of particular interest for reverse engineering, is different in the sense that the input domain is itself discrete. The mesh is often highly irregular and non-uniform, since it generally comes as the output of a surface reconstruction algorithm applied to a point cloud obtained from a scanning device.

Isotropic sampling leads to well-shaped triangles, and thus high-quality meshes when the notion of quality is related to the shape of the triangles. Such meshes are important for simulations where the quality of the mesh elements is critical. For digital geometry processing [3], most scanned models must undergo complete remeshing before processing. Many geometry processing algorithms (*e.g.*, smoothing, compression) benefit from isotropic remeshing, combined with uniform or curvature-adapted sampling.

1.1 Related Work

Parameterization-based remeshing techniques [4,5,6] have benefited from recent renewed interest in efficient parameterization methods for surface meshes [7, 8, 9, 10]. Here, the key is to parameterize the original mesh to obtain a bijective mapping and minimize the distortion due to the flattening process. The sampling and meshing stages are then considerably simpler on the (planar) parameter space. This allows both undersampling and oversampling with a high level of control by the user. Despite their recent popularity, these remeshing techniques (so-called "global approaches") have many drawbacks:

- **surface cutting:** each patch should be homeomorphic to a disk, therefore, closed or genus> 0 models have to be either cut along a cut graph to extract the polygonal schema [11], or decomposed into an atlas [7]. Finding a "smart" cut graph is not only known to be a delicate procedure [12, 13, 5], but also introduces a set of artificial boundary curves, associated pairwise. These boundaries, sampled as a set of curves (*i.e.*, 1-manifolds, while the surface has to be sampled as a 2-manifold), generate a visually displeasing seam tree. Some authors propose to apply a local mesh adaptation process to hide the seam a posteriori [6] but this solution is not fully satisfactory. Another solution to reducing the influence of the seam [10] consists of computing a globally smooth parameterization by decomposing the surface into patches and minimizing the distortion simultaneously across all patches. Although elegant, the latter solution does not remove the need for handling the patch boundaries during a global sample partitioning process.

- **parameterization and overlapping:** instead of constraining the user to fix the boundary onto a predetermined convex polygon, two recent methods minimize the distortion due to the parameterization by letting the solver find the "best" boundary while solving a linear system [8, 7]. Even though the gain in term of distortion is obvious, this approach does not solve the *overlapping* issues, contrary to other methods that may introduce additional seams [13] or generate an atlas [14].

- **numerical issues:** despite recent efforts for efficient computation of global parameterizations [7], the latter remains a time-consuming process for large models. Moreover, models with bad isoperimetric properties (*e.g.*, sock-like shapes) are numerically intractable for most state-of-the-art techniques.

- **lack of guarantees:** the conformal parameterization [15, 16, 8, 7] has often been the method of choice for irregular surface remeshing, isotropic [4, 6] or anisotropic [17]. Unfortunately, there exist triangulations for which this parameterization is not valid (see [18]), even when the boundary is fixed to be convex. Although the triangulation can be enriched by vertex insertion to produce a valid embedding, it is still unclear how many additional vertices are needed and what the guarantees are when using a scheme with a free, possibly concave, boundary.

The main alternative to global parameterization is known as the *mesh adaptation* process. It consists of performing a series of local modifications directly on the mesh, in embedding space. Remeshing algorithms using this approach [19, 20, 21, 22, 23, 24] usually involve computationally expensive optimizations in 3D. To improve efficiency, Frey and Borouchaki [22] use a less accurate optimization in the tangent plane. In a subsequent work, Frey [23] uses a paraboloid to obtain better approximation. The main issue of this approach is the fact that the mesh vertices must remain on the original mesh during the adaptation process. Otherwise, fidelity is quickly lost because of error accumulation. To solve this problem, the new optimal vertex positions are projected back to the original surface. Projecting the vertex involves a complex, computationally expensive and inaccurate computation that may even lead to topological errors during the remeshing process.

2 Main Contributions and Overview

In light of the drawbacks listed in the previous section, our main contribution is to combine the mesh adaptation process with a set of local, overlapping parameterizations. This allows us to handle large meshes of arbitrary genus. Another motivation of this paper is to formulate the issue of isotropic surface sampling using the concept of centroidal Voronoi tessellation. This way we shift from the so-called *unit length* paradigm used for numerical analysis [25] to the *unit cell* tiling paradigm, well suited for our application, *i.e.*, sampling of 2-manifolds. The first technique aims at generating meshes with unit edge length measured in a control space metric, while our algorithm aims to partition the surface with unit density *integrated* over the cells of a centroidal Voronoi tessellation. In particular, we show how the latter property is directly related to the notion of *isotropic* surface sampling.

Our technique uses two meshes: one is the piecewise smooth geometric reference, which we call the *geometric mesh* $\mathcal{M}_O$ (see Section 3). The second mesh $\mathcal{M}$ is initialized with a copy of the original mesh and evolves during the remeshing process until the desired mesh properties are achieved. Our technique falls into the category of local adaptation methods since remeshing is performed by a series of well-known local modifications: *edge-flip*, *edge-collapse*, *edge-split* and *vertex relocation*. The modifications are always applied sequentially to achieve desirable mesh characteristics.

The technique has three main stages: *complexity adjustment*, vertex *partitioning* and precise vertex *placement*. The first stage achieves the required number of vertices by applying iterative mesh simplification or refinement on the evolving mesh (see Section 4). The second stage uses a novel area-based remeshing technique to approximately partition the vertices in accordance with a density function specified on the original mesh (see Section 4). The second stage performs a precise isotropic placement of the vertices by constructing a weighted centroidal Voronoi tessellation (see Section 5). Section 7 shows some experimental results and Section 8 concludes.

3 Geometric Background

The input to our remeshing scheme is a 2-manifold triangle mesh $\mathcal{M}_O$ of arbitrary genus, possibly with boundaries. We consider $\mathcal{M}_O$ to be a piecewise linear approximation of a smooth surface, which is $\mathbf{C}^1$-continuous except at boundaries and a set of curves specified by feature edges. These feature edges can be provided by the user or computed automatically by feature detection techniques [26].

Surface reconstruction requires normal information at the mesh vertices. If the normals at the mesh vertices are not given, we use a method similar to [27, 24] to generate them: Every vertex is assigned a normal which is the weighted average of the normals of the faces adjacent to it. The weights are proportional to the angles of the corresponding faces at the vertex and sum to unity. Normals of a vertex lying on feature edges are not the same within all its adjacent faces. They are also defined by the weighted average of the face normals but as if the mesh was cut along the feature edges at the vertex.

3.1 Surface Approximation

Similarly to [28], we perform an estimate of the smooth surface in the vicinity of a mesh triangle. This may be obtained by reconstructing an approximation of the surface using triangular cubic Bézier patches for every face of $\mathcal{M}_O$. Vlachos *et al.* [29] presented a simple and efficient, yet robust and accurate, method to construct such curved patches called *PN triangles*. The triangle vertex normals together with vertex coordinates are used to construct a PN triangle. PN triangles usually (but not always) maintain a $\mathbf{G}^1$-continuous surface along adjacent triangles when their common vertices have identical normals. The normal of any point within a PN triangle is defined as a quadratic interpolation of the normals at the triangle vertices. Although Walton and Meek [30] presented a more complex and computationally expensive method to create triangular patches that guarantees

$\mathbf{G}^1$-continuity on the patch boundaries, we use PN triangles as a good tradeoff between accuracy and efficiency. Given a point q inside a triangular face $f = (q_1, q_2, q_3)$, the corresponding point on the surface of the PN triangle of f, as well as the normal at this point, can be uniquely defined by the barycentric coordinates of q with respect to f.

3.2 Controlling Fidelity

Our remeshing scheme performs a series of local mesh modifications. To ensure fidelity of the new mesh to the geometry of the original mesh, two measures are used to evaluate the distance between the two meshes. These measures are evaluated for every local modification on the region of the mesh affected by the modification. The modification is applied only if it does not violate the error conditions defined by the measures. The measures we use are conceptually similar to those of Frey and Borouchaki [22] are defined for a face instead of an edge. These measures were formulated in [28]. We briefly describe the measures and discuss their advantages.

Let $f = (v_1, v_2, v_3)$ be a face whose error is to be estimated. The first measure E_{smth} captures the degree of smoothness and should not exceed some threshold angle θ_{smth}:

$$E_{smth}(f) = \max_{i \in \{1,2,3\}} \langle N_f, N_{v_i} \rangle < \cos\theta_{smth}. \tag{1}$$

where N_f and N_v are unit normals of f and its vertex v, respectively; $\langle \cdot, \cdot \rangle$ denotes the dot product. N_v is taken from the original surface. Intuitively, E_{smth} describes how well f coincides with tangent planes of the surface at the vertices of f. The second measure E_{dist} captures the distance between f and the surface:

$$E_{dist}(f) = \max_{i \in \{1,2,3\}} \langle N_{v_i}, N_{v_{i+1}} \rangle < \cos\theta_{dist}. \tag{2}$$

Vertex indices are modulo 3; θ_{dist} is a threshold angle. A larger value of the maximal angle between the normals of two face vertices corresponds to a more curved surface above face f, and thus, to a greater distance. The beauty of these two measures is that they involve only normal directions. In addition to their computational efficiency, when used together, these two measures are also robust and accurate.

4 Initial Vertex Partition

To achieve the target mesh complexity, we apply local refinement or simplification operations. We perform a series of edge-collapse or vertex-split operations until the required number of vertices is achieved. Edges whose faces have minimal/maximal error metrics are simplified/refined first.

The heart of our remeshing scheme is the construction of the weighted centroidal Voronoi tessellation on the 3D mesh to achieve precise vertex placement (see Section 5). However, being optimal both in terms of sampling and isotropy, generating the weighted centroidal Voronoi tessellation is an extremely slow iterative process. This process first brings the mesh to the required sampling prescribed by a density function, then the mesh isotropy is optimized. It turns out that the first stage of the process is even slower than the second one, in contrast to many other iterative processes. The reason is that the process inherently maintains the local isotropy during resampling. To accelerate this process we first generate a coarse, initial sample partition by using a novel efficient area-based relaxation technique.

Alliez *et al.* [6] introduced an algorithm based on error diffusion that efficiently finds a good initial sampling. Unfortunately, this algorithm cannot guarantee fidelity of the resulting mesh to the original. Features that are not specified explicitly may be easily lost by this algorithm. In order to guarantee the mesh fidelity of the initial sampling we use an "area-based remeshing" technique, which is based on a series of local mesh modifications, while validating the mesh fidelity by the error measure described in Section 3.2.

The area-based remeshing technique was first introduced by Surazhsky and Gotsman [31, 28]. It is based on the idea of locally equalizing the area of triangles or bringing the areas to the ratios specified by the density function. After this, it remains to achieve a precise isotropic vertex placement.

5 Precise Vertex Placement

Our goal is to isotropically sample a density function specified on the original surface mesh $\mathcal{M}_O$. There are, thus, two terms (sampling and isotropy) to be defined:

- Sampling: to partition a density function among a set of samples. The density function is defined over a bounded domain, which must be partitioned so that we obtain a *tiling*, or tessellation, where each tile corresponds to exactly one sample, without overlapping or holes. The density partition must be done so that we obtain the so-called equal-mass enclosing property, namely, each tile contains the same amount of density.
- Isotropy: the shape of each tile is not biased with respect to any particular direction. In other words, each cell is as compact (*i.e.*, as "round") as possible. In the uniform case the ideal tile is a disk, which maximizes the compactness, but does not produce a tiling of the domain. The hexagonal lattice better conforms with *uniform* tiling along with optimal compactness. The *non-uniform* case leads to a tradeoff between compactness and partition of the density function.

5.1 Centroidal Voronoi Tessellation

The initial triangulation gives us a vertex partition, which defines a tiling of a 2D parameter space. Each triangular tile corresponds to three samples (the vertices of the triangle) instead of one as desired. We, therefore, use the dual of the triangulation, *i.e.*, the tessellation in which each tile is now associated with exactly one sample. We aim at obtaining a special class of Voronoi tessellations, the so-called *centroidal Voronoi tessellation*, with the two properties mentioned above, *i.e.*, equal-mass enclosing and isotropy.

Given a density function defined over a bounded domain Ω, a weighted centroidal Voronoi tessellation [32] (denoted WCVT) of Ω is a class of Voronoi tessellations, where each site coincides with the centroid (*i.e.*, center of mass) of its Voronoi region. The centroid c_i of a Voronoi region V_i is calculated as:

$$c_i = \frac{\int_{V_i} x\rho(x)dx}{\int_{V_i} \rho(x)dx} \tag{3}$$

where $\rho(x)$ is the density function. This structure turns out to have a surprisingly broad range of applications for numerical analysis, location optimization, optimal partition of resources, cell growth, vector quantization, etc. (see [32]). This follows from the mathematical importance of its relationship with the energy function

$$E(z, V) = \sum_{i=1}^{n} \int_{V_i} \rho(x)|x - z_i|^2 dx \tag{4}$$

where $V \in \Omega$ and $z \in V$. It is proven in [33] that (i) the energy function is minimized at the mass centroid of a given region, and (ii) for a given set of centers $Z = \{z_i\}$, the energy function $E(Z, V)$ is minimized when V is a Voronoi tessellation.

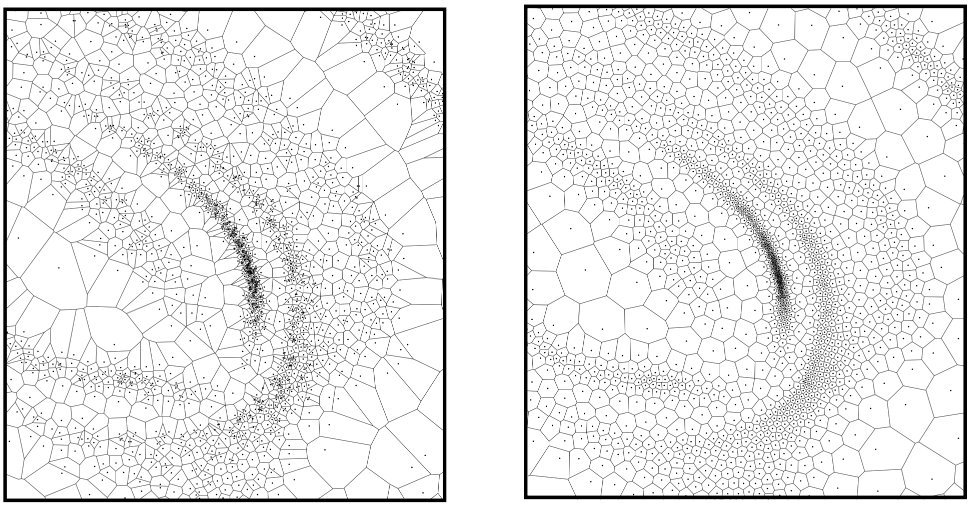

Figure 1: Left: Ordinary Voronoi tessellation of a point set sampled from some density function. Right: Point set and its corresponding weighted centroidal Voronoi tessellation for the same density function . Each site coincides with the center of mass of its Voronoi cell. The sample set on the right was generated by Lloyd iterations applied to the sample set on the left.

5.2 Building a WCVT on a 3D Mesh

One way to build a weighted centroidal Voronoi tessellation is to use Lloyd's relaxation method. The Lloyd algorithm is a deterministic, fixed point iteration [34]. Given a density function and an initial set of n sites, it consists of the following three steps:

1. Construct the Voronoi tessellation corresponding to the n sites;
2. Compute the centroids of the n Voronoi regions with respect to the density function expressed in local parameter space, and move the n sites to their respective centroids;
3. Repeat steps 1 and 2 until satisfactory convergence is achieved.

Since a Delaunay triangulation and its corresponding Voronoi tessellation are dual, we do not need to work explicitly with a Voronoi tessellation but rather with its dual triangulation. We adapt the Lloyd algorithm in the following manner. Instead of constructing the Voronoi tessellation for the point set of the current mesh, we modify the mesh by a series of Delaunay edge flips in order to maintain the *local* Delaunay property of the mesh. For every vertex, we then compute its Voronoi cell in a local parametric domain, and move the vertex to the new 3D location corresponding to the centroid of the cell. We now describe these steps in detail.

Updating the local Delaunay property Notice that the usual definition of the Voronoi tessellation holds for a set of sites in Euclidean space, *i.e.*, in the 2D plane for partitioning a 2-manifold. As demonstrated in [35], Voronoi diagrams can also be constructed in Riemannian manifolds for sufficiently dense sets of points. In our algorithm, the current 3D triangulation is the result of a series of local mesh adaptations performed for initial vertex partition. Each local mesh adaptation has been performed while maintaining a "local" 2D Delaunay property. Instead of building a new Voronoi tessellation at each step of the Lloyd relaxation process, we *restore* the local Delaunay property by performing a series of edge flips in 3D. This maximizes the smallest angle property. This task is performed efficiently by updating a priority queue sorted by the angles.

Computing the centroid Every relaxation step in the sequence of Lloyd iterations moves a vertex v from the newly generated mesh to the centroid of its "Voronoi" cell (we abuse the word Voronoi here, since the cell is not planar or even convex). To proceed we first need to define a *planar* Voronoi cell for v. Denote the vertices incident to v as $v_1, \ldots, v_k$, where k is the degree of v. Let $\mathcal{S}(v)$ be a sub-mesh of $\mathcal{M}$ containing only v, $v_1, \ldots, v_k$ and faces incident on v. We reduce the problem in 2D by mapping $\mathcal{S}(v)$ onto the plane using a natural and simple method approximating the geodesic polar map [36], described by Welch and Witkin [37] and later by Floater [38]. The method preserves the lengths of edges incident to v, and the relative angles of $\mathcal{S}(v)$ at v. This method is an efficient and precise approximation of a conformal mapping of $\mathcal{S}(v)$ onto the plane. Let p, $p_1, \ldots, p_k$ be the positions of vertices v, $v_1, \ldots, v_k$ within the resulting mapping $\mathcal{S}_P(v)$. p is mapped to the original. Then we construct a Voronoi cell of v in $\mathcal{S}_P(v)$ with respect to the circumcenters of the triangles built from p and $p_1, \ldots, p_k$, and compute the centroid p_{new} of this cell with respect to an approximation of the density function specified over the original mesh. The latter approximation consists of evaluating the density function at the new mesh vertices and piecewise linearly interpolating the resulting density over the new mesh triangles.

5.3 Vertex Relocation

Knowing the new vertex position of v (p_{new}), we need to bring it back to the original surface of the given mesh, namely, to find the position of v denoted by $\mathbf{x}_{new}(v)$ on $\mathcal{M}_O$ that corresponds to p_{new}. Existing remeshing methods, *e.g.*, [20, 22, 24] solve this problem by computing the vertex projection onto the original surface. As stated in Section 1.1, projecting the vertex involves an expensive and possibly inaccurate computation that may even lead to topological errors. We solve this problem using a mesh parameterization with low distortion and guarantee of bijective mapping. This way we can deduce $\mathbf{x}_{new}(v)$ precisely and efficiently.

We now briefly describe how to find $\mathbf{x}_{new}(v)$ using mesh parameterization. For every vertex of $\mathcal{M}$ we maintain its exact position on the original surface using barycentric coordinates of the vertex within a specific face of $\mathcal{M}_O$. Note that this gives us a unique point on the reconstructed surface defined by PN triangles over $\mathcal{M}_O$; see Section 3.1. The central idea in using parameterization to locate $\mathbf{x}_{new}(v)$ is to use barycentric coordinates of p_{new} with respect to a face of $\mathcal{S}(v)$ that contains it. Using these barycentric coordinates together with the barycentric coordinates of the face vertices, we locate a point in the parametric domain of $\mathcal{M}_O$. This point is then elevated to the original surface.

However, this simple scheme is only applicable when we have a well-defined parametric domain embedded in the 2D plane. Since not all 3D meshes are isomorphic to a disk, such a 2D parametric domain may not exist. To solve this problem we use a novel dynamic patch-wise parameterization technique introduced independently by Vorsatz *et al.* [39] and Surazhsky and Gotsman [28]. This technique aims to overcome the problems of global parameterization (see Section 1.1) and allows the handling of meshes of arbitrary genus and boundaries. It maintains a set of small (usually manifold) overlapping patches and their corresponding conformal parameterization. Every patch is constructed on demand depending on a specific local modification and contains the region required to locate a new vertex position in the 2D parametric domain. Reuse of the patches already parameterized guarantees the efficiency of this technique both in terms of computational cost and memory consumption. See Figure 2 which demonstrates how this technique is used for vertex relocation.

6 Preserving Features

Note that near feature creases and boundaries, the computation of the centroid must be more sophisticated. To proceed we *clip* the Voronoi cells with the set of feature edges [40]. This allows us to disconnect two smooth regions separated by a feature crease during the computation of the centroid. It leads to a nice sampling quality in the vicinity of the features, obtained through the non-symmetric behavior of the algorithm (the feature edges influence the surface samples but the surface samples do not influence the samples on a feature edge). At the intuitive level, two samples adjacent in the Voronoi tessellation and separated by a feature do not influence each other, and the samples close to a feature edge are repulsed by the constraints (see Figure 3). Geometrically, clipping a cell by the set of constraints removes some regions from the computation of the centroid, making the Lloyd

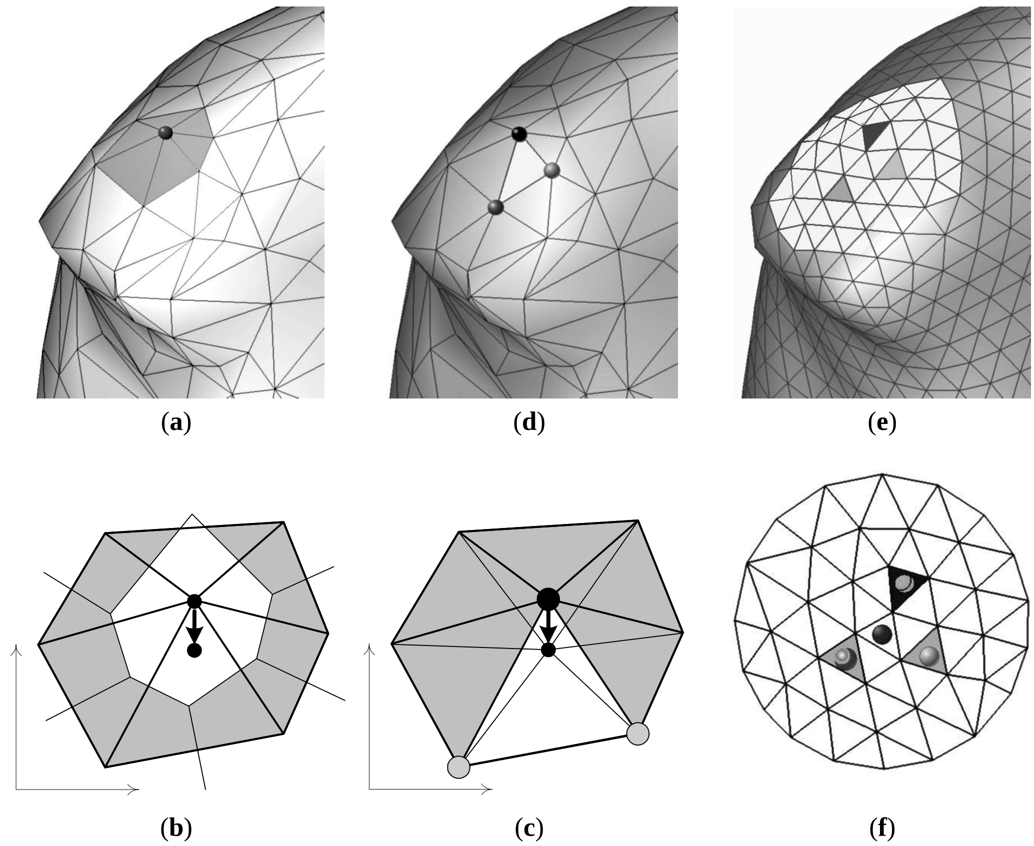

Figure 2: Vertex relocation. (**a**) A vertex v of the mesh to be relocated. The faces of the sub-mesh $\mathcal{S}(v)$ are dark-grey. (**b**) $\mathcal{S}(v)$ is mapped onto the plane and the Voronoi cell of v is constructed. The new vertex location is a weighted centroid of the cell. (**c**) The triangle containing the new location. (**d**) The corresponding triangle in the mesh $\mathcal{M}$. (**e**) The highlighted vertices in (d) correspond to three faces of the original mesh $\mathcal{M}_O$. (**d**) A patch containing all these faces of $\mathcal{M}_O$ is constructed and then parameterized. The new location of v in the patch is computed using the corresponding barycentric coordinates of the 2D mapping (c).

relaxation consistent with respect to the features. Once the centroid has been computed, it remains to relocate the vertex v to the centroid.

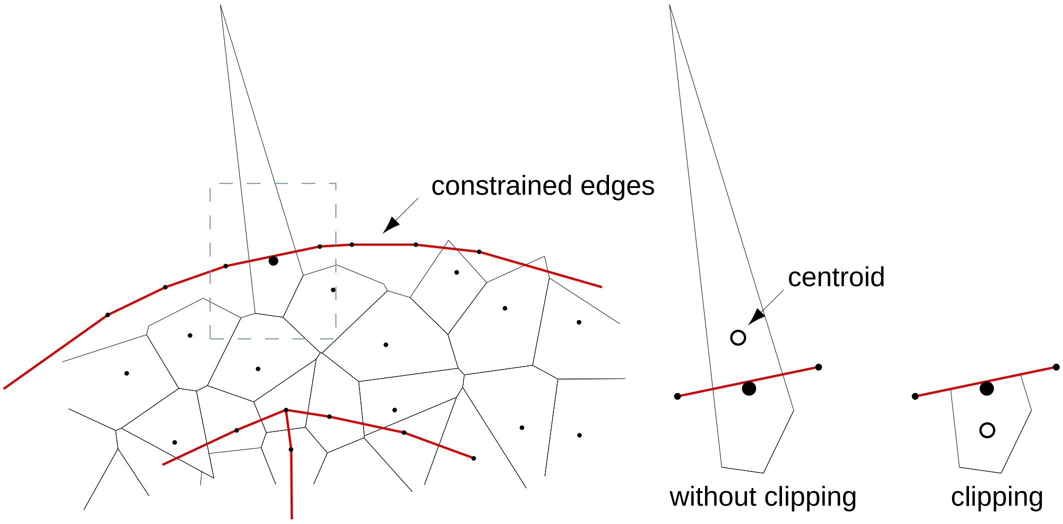

Figure 3: Left: A Voronoi tessellation in parameter space with a feature skeleton. All the cells are drawn according to the circumcircle property. Computing the centroid without clipping by the constraints makes the sampling inconsistent, while the effect of clipping is to repulse the samples from the boundary or sharp edges, the centroid being computed on the truncated cell. A constrained edge separating two samples thus acts as a barrier [40] annihilating their mutual influence.

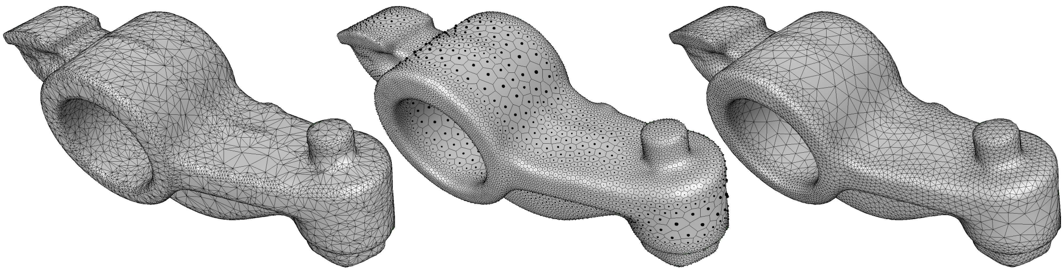

Figure 4: Left: original. Middle: centroidal tessellation. Right: curvature-adapted remeshing.

7 Experimental Results

The algorithm described in this paper has been implemented in an interactive software package. Similarly to [28], the user can control the remeshing via the definition of the density function, either

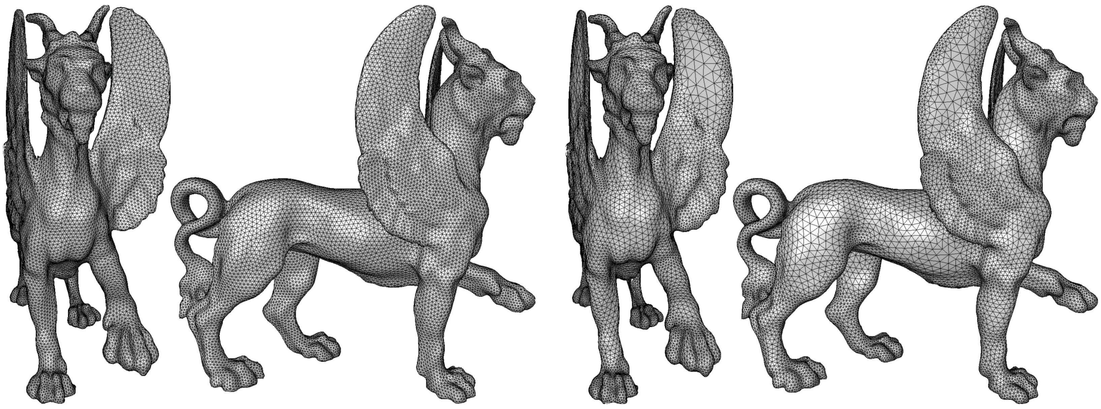

Figure 5: Left: uniform remeshing. Right: curvature-adapted remeshing.

uniform, or adapted to the curvature. We also provide an option to smooth the density function and therefore obtain a smoother mesh gradation.

We have run our technique on a variety of models of arbitrary genus and complexity. Figure 4 illustrates a curvature-adapted remeshing of the *rocker-arm* model with 10,000 vertices (the same as the original model). The tessellation shown in the middle is drawn by tracing an edge between the circumcenters of two incident triangles, every circumcenter located in the support plane of the corresponding triangle. The genus 1 *feline* model is remeshed both uniformly and with a curvature-based density. The original model of 50,000 vertices was first simplified to 20,629 vertices by local mesh adaptation. The initial sampling using area-based remeshing required only 8 iterations. To obtain the same resampling using just the lloyd procedure required 45 iterations. Note also that every area-based iteration that relocates each of the mesh vertices is about twice as fast as a Lloyd iteration. The polishing of the isotropy then took 15 Lloyd iterations. The entire remeshing was performed in less than two minutes on a Pentium 4 2.4GHz machine with 512MB of memory. Figure 6 shows a uniform remesh of the piecewise smooth model *fandisk*, containing 5,000 vertices. The helmet, a genus 3 model, is remeshed with a curvature-related density function (see Figure 7). Figure 9 illustrates a uniform remeshing of the David model, part of the digital Michelangelo project [41]. The irregular and non-uniform input mesh contains 350,000 vertices, while the remeshed model has 100,000 vertices. The initial vertex partition stage runs for 5.5 minutes, and the vertex placement runs for 4 minutes. We chose this model for illustrating the scalability and the adaptability of our technique to handle both complex models and arbitrary genus. Figure 8 shows a closeup of the same model to emphasis the quality of sampling obtained by centroidal tessellation.

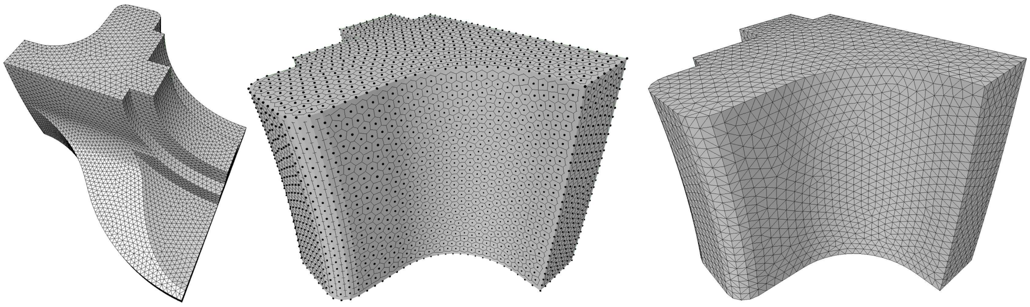

Figure 6: Uniform remeshing of the fandisk model (piecewise smooth).

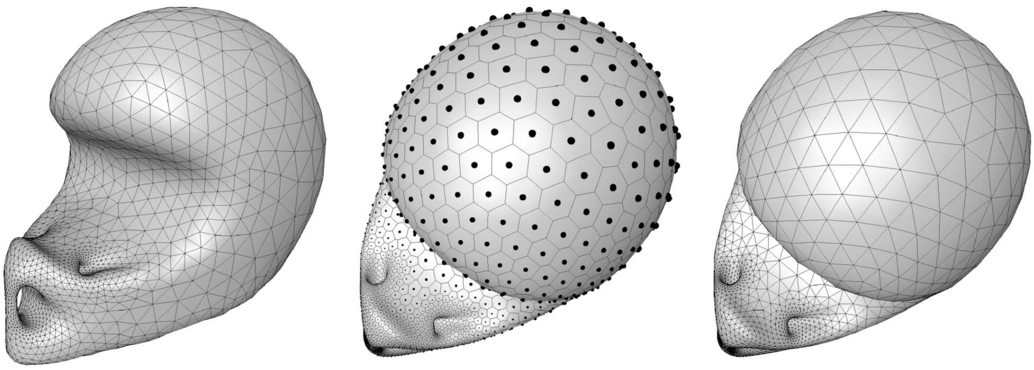

Figure 7: Curvature-adapted remeshing of the helmet model (genus 3).

8 Conclusion

This paper has introduced a technique for efficient and precise isotropic surface remeshing. Our approach first performs efficient sampling of the mesh with respect to a density function using the area-based remeshing technique. A Lloyd relaxation stage that constructs a weighted centroidal Voronoi tessellation is then directly applied on the mesh to ensure precise isotropic placement of the vertices. Using a patch-wise parameterization technique to apply a local 2D Lloyd relaxation on the 3D mesh allows us to handle complex models with arbitrary genus and any number of boundaries. Thus, by combining state-of-the-art techniques we are able to efficiently produce high quality isomorphic remeshings. One limitation of our method is the convergence behavior of the Lloyd relaxation process for precise isotropic vertex placement. As explained in [32], local convergence is guaranteed in 2D when the density function is log concave. Since in our case the density function is either uniform when requested, or a function of the curvature, this does not guarantee the local convergence in all cases. Nevertheless, it was not an issue in our experiments. As future work we plan to accelerate further the Lloyd relaxation.

References

[1] M. Bern and D. Eppstein. Mesh Generation and Optimal Triangulation. *Computing in Euclidean Geometry, edited by D.-Z. Du and F. K. Hwang*, pages 23–90, 1992.

[2] Marshall Bern and Paul Plassmann. Mesh generation. In J. Sack and J. Urrutia, editors, *Chapter 6 in Handbook of Comp. Geometry*. Elsevier Science, 1999.

[3] Wim Sweldens and Peter Schröder, editors. *Digital Geometry Processing*. Course Notes. ACM SIGGRAPH, 2001.

[4] Pierre Alliez, Mark Meyer, and Mathieu Desbrun. Interactive geometry remeshing. *ACM Transactions on Graphics*, 21(3):347–354, 2002. SIGGRAPH conference proceedings.

[5] X. Gu, S. Gortler, and H. Hoppe. Geometry Images. In *Proceedings of SIGGRAPH*, pages 355–361, 2002.

[6] Pierre Alliez, Éric Colin de Verdière, Olivier Devillers, and Martin Isenburg. Isotropic surface remeshing. In *Proceedings of Shape Modeling International*, 2003.

[7] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least Squares Conformal Maps for Automatic Texture Atlas Generation. In *Proc. of SIGGRAPH*, pages 362–371, 2002.

[8] Mathieu Desbrun, Mark Meyer, and Pierre Alliez. Intrinsic parameterizations of surface meshes. In *Proceedings of Eurographics*, pages 209–218, 2002.

[9] P. Sander, S. Gortler, J. Snyder, and H. Hoppe. Signal-specialized parametrization. In *Eurographics Workshop on Rendering 2002*, 2002.

[10] Andrei Khodakovsky, Nathan Litke, and Peter Schröder. Globally Smooth Parameterizations with Low Distortion. In *Proceedings of SIGGRAPH*, 2003. To appear.

[11] F. Lazarus, M. Pocchiola, G. Vegter, and A. Verroust. Computing a Canonical Polygonal Schema of an Orientable Triangulated Surface. In *Proc. of 17th Annu. ACM Sympos. Comput. Geom.*, pages 80–89, 2001.

[12] Jeff Erickson and Sariel Har-Peled. Optimally cutting a surface into a disk. In *Proceedings of the 18th Annual ACM Symposium on Computational Geometry*, pages 244–253, 2002.

[13] A. Sheffer. Spanning Tree Seams for Reducing Parameterization Distortion of Triangulated Surfaces. In *Proceedings of Shape Modeling International*, 2002.

[14] Olga Sorkine, Daniel Cohen-Or, Rony Goldenthal, and Dani Lischinski. Bounded-distortion Piecewise Mesh Parameterization. In *IEEE Visualization Conference Proceedings*, 2002.

[15] Ulrich Pinkall and Konrad Polthier. Computing Discrete Minimal Surfaces and Conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[16] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution Analysis of Arbitrary Meshes. In *Proceedings of SIGGRAPH*, pages 173–182, 1995.

[17] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Levy, and Mathieu Desbrun. Anisotropic polygonal remeshing. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, 2003. To appear.

[18] Kai Hormann. From Scattered Samples to Smooth Surfaces. In *Proceedings of Geometric Modeling and Computer Graphics*, 2003.

[19] Greg Turk. Re-tiling polygonal surfaces. In *Computer Graphics. SIGGRAPH '92 Proceedings*, volume 26, pages 55–64, 1992.

[20] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *Computer Graphics. SIGGRAPH '93 Proceedings*, volume 27, pages 19–26, 1993.

[21] Hugues Hoppe. Progressive meshes. In *Computer Graphics. SIGGRAPH '96 Proceedings*, volume 30, pages 99–108, 1996.

[22] Pascal J. Frey and Houman Borouchaki. Geometric surface mesh optimization. *Computing and Visualization in Science*, 1:113–121, 1998.

[23] Pascal J. Frey. About surface remeshing. In *Proceedings of 9th International Meshing Roundtable*, pages 123–136, 2000.

[24] A. Rassineux, P. Villon, J-M. Savignat, and O. Stab. Surface remeshing by local Hermite diffuse interpolation. *International Journal for Numerical Methods in Engineering*, 49:31–49, 2000.

[25] P. L. George and H. Borouchaki, editors. *Delaunay Triangulation and Meshing Application to Finite Elements*. Hermes, Paris, 1998.

[26] K. Watanabe and Alexander Belyaev. Detection of salient curvature features on polygonal surfaces. *Computer Graphics Forum*, 20(3):385–392, 2001.

[27] Steven J. Owen, David R. White, and Timothy J. Tautges. Facet-based surfaces for 3D mesh generation. In *Proc. of 11th International Meshing Roundtable*, pages 297–312, 2002.

[28] Vitaly Surazhsky and Craig Gotsman. Explicit surface remeshing. In *Proceedings of Eurographics Symposium on Geometry Processing*, pages 17–28, June 2003.

[29] Alex Vlachos, Jorg Peters, Chas Boyd, and Jason L. Mitchell. Curved PN triangles. In *Symposium on Interactive 3D Graphics*, pages 159–166, 2001.

[30] D.J. Walton and D.S. Meek. A triangular G^1 patch from boundary curves. *Computer Aided Design*, 28(2):113–123, 1996.

[31] Vitaly Surazhsky and Craig Gotsman. High quality compatible triangulations. In *Proceedings of 11th International Meshing Roundtable*, pages 183–192, Ithaca, NY, September 2002.

[32] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal Voronoi Tesselations: Applications and Algorithms. *SIAM review*, 41(4):637–676, 1999.

[33] Roger Brauwerman, Sarah Joy Zoll, Christopher L Farmer, and Max Gunzburger. Centroidal Voronoi Tessellations Are Not Good Jigsaw Puzzles, 1999. preprint, found at http://www.math.iastate.edu/reu/1999/cvt.pdf.

[34] S. Lloyd. Least square quantization in PCM. *IEEE Trans. Inform. Theory*, 28:129–137, 1982.

[35] Greg Leibon and David Letscher. Delaunay triangulations and voronoi diagrams for riemannian manifolds. In *Proceedings of the sixteenth annual symposium on Computational geometry*, pages 341–349, 2000.

[36] Dirk Jan Struik. *Lectures on Classical Differential Geometry*. Dover Publications, Inc., 1988.

[37] William Welch and Andrew Witkin. Free form shape design using triangulated surfaces. In *Computer Graphics (Proc. SIGGRAPH '94)*, volume 28, 1994.

[38] Michael S. Floater. Parameterization and smooth approximation of surface triangulation. *Computer Aided Geometric Design*, 14:231–250, 1997.

[39] Jens Vorsatz, Christian Rössl, and Hans-Peter Seidel. Dynamic remeshing and applications. In *Proceedings of the 8th Conference on Solid Modeling and Applications*, June 2003.

[40] A. Lingas. Voronoi diagrams with barriers and the shortest diagonal problem. *Inform. Process. Lett.*, 32:191–198, 1989.

[41] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital michelangelo project. In Kurt Akeley, editor, *ACM SIGGRAPH Conference Proceedings*, pages 131–144, 2000.

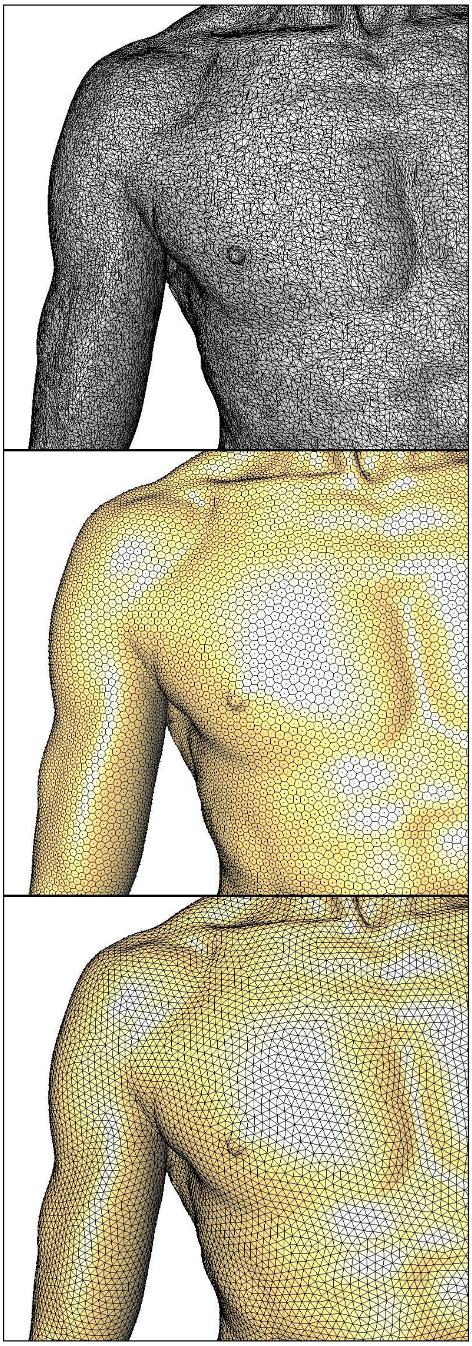

Figure 8: Closeup on the Digital Michelangelo David model: original, uniform sample tiling and triangle remesh.

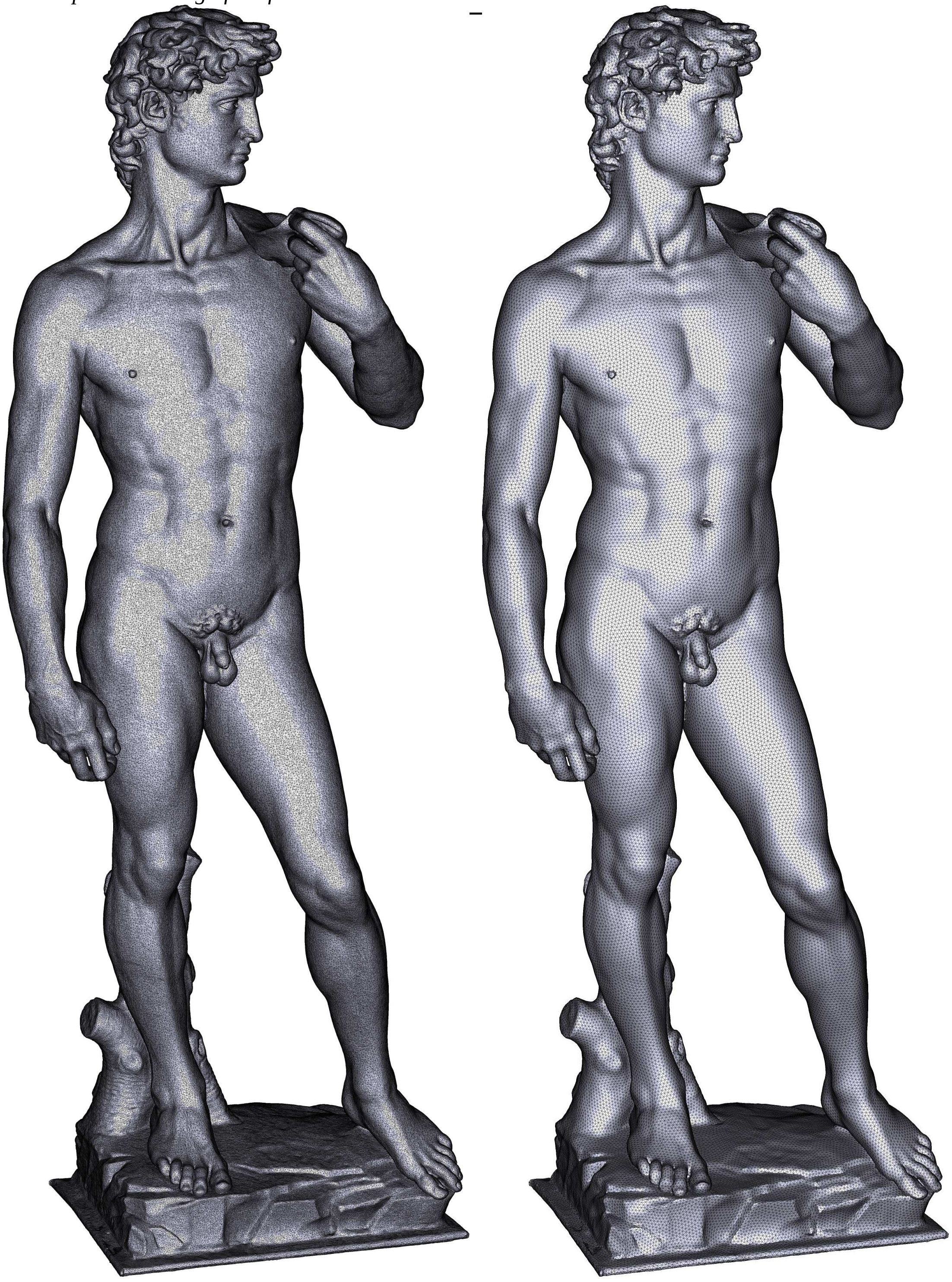

Figure 9: Left: Digital Michelangelo David model (350k vertices). Right: uniform remeshing (100k vertices).

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifi que
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399